Remarks

In the Office Action, the Examiner noted that claims 1-63 are pending in the application, and that claims 1-63 are rejected. By this amendment, claims 16 and 60 have been amended. Thus, claims 1-63 are pending in the application.

Applicant hereby requests further examination and reconsideration of the application, in view of the foregoing amendments.

In the Specification

The Title has been amended as required by the Examiner in a previous Office Action.

In the specification, the table on page 1 has been amended to identify the co-pending applications by their serial numbers.

In the Claims

Rejection Under 35 USC 103

The Examiner rejected claims 1-6, 10-33, 36-53, and 57-63 under 35 U.S.C. § 103(a), as being unpatentable over *Stiles, et al.*, U.S. Patent No. 5,163,140 (hereinafter *Stiles*) in view of *Gochman, et al.*, U.S. Patent No. 5,964,868 (hereinafter *Gochman*).

The Examiner rejected claims 7, 34-35, and 54-56 under 35 U.S.C. § 103(a), as being unpatentable over *Stiles* in view of *Gochman* and *Fite et al.*, U.S. Patent No. 5,142,634.

The Examiner rejected claims 8 and 9 under 35 U.S.C. § 103(a), as being unpatentable over *Stiles* in view of *Gochman* and *Brown et al.*, U.S. Patent No. 5,964,868.

Applicant respectfully traverses the rejection of claims 1-63.

Claims 16-45

With respect to claim 16, the Examiner asserts that *Stiles* teaches control logic, coupled to said BTAC, configured to control a multiplexer to select said speculative target address as said fetch address during a first period whether or not said previously executed branch instruction is present in said line. Applicant respectfully asserts that *Stiles* does not teach control logic configured to control a multiplexer to select a speculative target address provided by a BTAC as a fetch address for selecting a line of instruction bytes selected

by the fetch address whether or not at least one branch instruction is present in the cache line, as recited by amended claim 16.

First, the Examiner cites text in *Stiles* (col. 13, line 47 to col. 14, line 2) that refers to a multiplexing network 256 of Fig. 8. The multiplexing network does not receive a speculative target address nor does it supply a fetch address to an instruction cache.

Second, Applicant can find no teaching in *Stiles* of control logic that selects a speculative target address as an instruction cache fetch address whether or not at least one branch instruction is present in a cache line selected by the fetch address. The Examiner cites col. 9, lines 27-58 in *Stiles*. The cited text states in relevant part: "The primary function of IDC 162 [instruction decode control circuitry] is to control decoding of instructions, and in particular to ensure the proper handling of branches based on information from the BPC. IDC 162 also determines where instructions can be fetched from and controls IFC 165 [instruction fetch control circuitry] accordingly. IFC 165 fetches from ICache 30 or main memory, generates fetch PC's, and provides a delayed version of the fetch PC as a decode PC to the first and second level BPC's." The remainder of the recited text describes the two-level structure of the BPC. Thus, Applicant respectfully asserts the text of *Stiles* cited by the Examiner does not teach control logic that selects a speculative target address as an instruction cache fetch address whether or not at least one branch instruction is present in a cache line selected by the fetch address, and Applicant can find no such teaching in *Stiles*.

In the Response to Arguments, the Examiner refers to col. 9, lines 58-62 and col. 16, line 59 to col. 17, line 27 in *Stiles*. Col. 9, lines 58-62 states: "In parallel with instruction decoding, the instruction's decode PC is used to perform parallel lookups in the first and second level BPC's. (Since the incoming instructions have not been decoded at this point, non-branch instructions are also checked.)" Col. 17, lines 7-8 state: "a second level cache look-up effectively is assumed to always hit." The Examiner correctly states that the second level branch prediction cache may predict a target address for a non-branch instruction.

However, this is not the limitation claim 16 recites. Claim 16 recites that the control logic *selects a speculative target address as an instruction cache fetch address* whether

or not at least one branch instruction is present in the cache line. *Stiles* does not explicitly teach that the processor selects the target address supplied by the second level branch prediction cache as the fetch address whether or not a branch instruction is present. Furthermore, it does not necessarily follow from the fact that the second level branch prediction cache provides a target address that the processor selects it as the fetch address whether or not a branch instruction is present. In other words, it is certainly possible that since the instruction at the decode PC is decoded in parallel with the second level branch prediction cache lookup, that the processor only selects the predicted target address as the fetch address if the instruction decoder indicates a branch instruction is present. Applicant asserts that *Stiles* does not clearly indicate either way; however, Applicant asserts the following teachings of *Stiles* imply that *Stiles*' processor only selects the predicted target address as the fetch address if it knows via decoding that a branch instruction is present.

1. At col. 5, line 11: "As *branch instructions* are decoded, the BPC is consulted for information *about that branch.*" This implies it is known that a branch instruction is present.
2. At col. 6, line 52 to col. 8, line 8: This text discusses various exception processing techniques used by *Stiles*' processor; however, it does not say what the exceptions are. In particular, it does not teach one of the possible exceptions being that the second level branch prediction cache predicted target address based on the decode PC that was selected as the fetch address, but subsequently it was determined that there was no branch instruction present at the decode PC.
3. At col. 9, line 36: The instruction decode control controls the instruction fetch control and determines where instructions can be fetched from. This implies fetching is based on decoding of branches, rather than branching without decoding to know whether a branch instruction is present.
4. At col. 16, line 53: "As each *branch instruction* is fetched its address is used to perform parallel look-ups in the two levels of BPC". This implies it is known that a branch instruction is present.
5. At col. 17, line 3 (see also col. 16, line 27): "In the case of a first level cache miss, the prediction information read out from the indexed second level cache entry is used to at least predict some aspects of the branch instruction's processing. ... a second level cache look-up effectively is assumed to always hit." The decode PC for every single instruction hits in the second level branch prediction cache, including non-branch instructions, since there aren't any valid bits to indicate that a previously executed branch was present at this decode PC, and there are no tags to match with the remainder of the decode PC. Consequently, to begin fetching and executing instructions at the target address predicted by the second level branch prediction cache if the direction was predicted taken without first knowing a branch instruction is present, would result in an unacceptable amount of

mispredictions. That is, *Stiles*' processor would branch for every single instruction, branch or non-branch, for which the selected second level branch prediction cache's direction bit indicates a taken branch, which would result in many mispredictions and pipeline flushes, thus degrading performance of the processor rather than improving it. To illustrate the magnitude of the mispredictions, assume branches are taken approximately 50% of the time, and assume approximately 20% of the instruction stream is branch instructions and 80% is non-branch instructions, both of which are common. Then the second level branch prediction cache would mispredict for 40% of all instructions fetched! However, *Stiles* teaches that his second level branch prediction cache hit rate is high. See col. 10, line 20: "To result in a net performance benefit the cache must provide ... prediction information ... for a high enough percentage of the branch instructions processed by the CPU." Thus, it is more reasonable to infer that *Stiles* knows whether there is a branch instruction present before deciding to select the predicted target address as the fetch address, and therefore does not teach the limitation recited by claim 16.

6. At col. 17, line 18: "There will still be a delay before processing of target instructions can begin, if the branch is predicted taken." Why? The text is not clear, but quite likely because *Stiles* has to wait and decode the instruction to verify that it is a branch instruction.
7. At col. 17, line 25: "whether the prediction was based on cached information from an earlier execution of this branch or from some other branch." This implies that at least it is known there is a branch instruction present. Specifically, the text does not say, "or from a non-branch," implying that a branch instruction is known to be present.

In summary, the reference is required to teach the claim limitation, not to teach something from which the limitation *may be inferred.* Furthermore, the reference does not teach the limitation inherently, since the possibility exists that *Stiles*' processor decodes the instruction to know that a branch instruction is present before deciding to select the predicted target address as the fetch address.

Furthermore, amended claim 16 recites prediction check logic that detects that the control logic controlled the multiplexer to select the speculative target address erroneously because zero branch instructions are present in the selected cache line. Applicant can find no teaching of this claim limitation is *Stiles*. Furthermore, as discussed above, the text at col. 6, line 52 to col. 8, line 8 of *Stiles* discusses various exception processing techniques used by *Stiles*' processor; however, it does not teach the prediction check logic determining zero branch instructions are present in the selected cache line, as recited in amended claim 16. Furthermore, the text at col. 16, lines 33-35 states: "Predicted target addresses must, of course, eventually be checked elsewhere within the

CPU." However, checking a predicted target address against a correct target address is not determining that zero branch instructions are present in the selected cache line, particularly since the incorrect predicted target address could be for a different branch instruction.

For the reasons stated above, Applicant respectfully asserts that *Stiles* in view of *Gochman* does not obviate claim 16.

With respect to claim 22, the Examiner asserts that *Stiles* teaches a location within a cache line of a previously executed branch instruction is cached in a BTAC. Applicant respectfully asserts *Stiles* does not teach a location within a cache line of a previously executed branch instruction cached in a BTAC, and in particular the text cited by the Examiner does not teach this limitation. *Stiles* specifically teaches the information cached in *Stiles*' first level BPC is: a target address, up to 24 bytes of an instruction stream starting at the target address, and two direction history bits; and the information cached in *Stiles*' second level BPC is: two bytes of a 4-byte target address and a single history bit. Col. 10, lines 43-48; col. 15, lines 33-35. For similar reasons, Applicant respectfully asserts *Stiles* does not teach the limitations recited in claims 26, 30, and 32.

With respect to claims 36 and 37, the text of *Stiles* cited by the Examiner does not teach the recited claim limitations, and Applicant can find no teaching of the limitations in *Stiles*.

Applicant respectfully asserts *Stiles* in view of *Gochman* does not obviate dependent claims 17-45 because they depend from independent claim 16, which is not obviated by *Stiles* in view of *Gochman* for the reasons discussed above.

Claims 1-13

With respect to claim 1, the Examiner asserts that *Stiles* has taught a storage element for storing an indication of whether the microprocessor branched to the speculative target address provided by the BTAC without knowing whether an instruction associated with the indication is a branch instruction, and prediction check logic for notifying branch control logic that the microprocessor erroneously branched to a speculative target address if instruction decode logic indicates the instruction is not a branch instruction and the

indication indicates the microprocessor branched to the speculative target address. Applicant respectfully asserts that *Stiles* has not taught either of these limitations of claim 1 for reasons similar to those discussed above with respect to claim 16. In particular, the text of *Stiles* cited by the Examiner does not teach the limitations, and Applicant can find no teaching of the limitations in *Stiles*. Although *Stiles*' second level branch prediction cache may predict a target address for a non-branch instruction, this is not the limitation claim 1 recites. Claim 1 recites that the microprocessor *branches* to the speculative target address provided by the BTAC without knowing whether an instruction associated with the indication is a branch instruction. However, *Stiles* does not explicitly teach the claim limitation; furthermore, *Stiles* does not teach the limitation inherently, since the possibility exists that *Stiles*' processor decodes the instruction to know that a branch instruction is present before branching to the predicted target address.

Applicant respectfully asserts *Stiles* in view of *Gochman* does not obviate dependent claims 2-13 because they depend from independent claim 1, which is not obviated by *Stiles* in view of *Gochman* for the reasons discussed above.

Claims 14-15 and 46-63

Applicant respectfully asserts *Stiles* in view of *Gochman* does not obviate independent claims 14, 46, 50, 54, 57, and 59-63 for the reasons discussed above with respect to claims 1 and 16.

Applicant respectfully asserts *Stiles* in view of *Gochman* does not obviate dependent claims 15, 47-49, 51-53, 55-56, or 58 because they depend from independent claims 14, 46, 50, 54, and 57, respectively, which are not obviated by *Stiles* in view of *Gochman* for the reasons discussed above.

Applicant respectfully requests that a timely Notice of Allowance be issued in this case.

Applicant earnestly requests the examiner to telephone him at the direct dial number printed below if the examiner has any questions or suggestions concerning the application.

Respectfully submitted,

E. Alan Davis

E. Alan Davis
Huffman Law Group, P.C.
Registration No. 3959,54
Customer No. 23669
1832 N. Cascade Ave.
Colorado Springs, CO 80907
719.475.7103
719.623.0141 fax
alan@huffmanlaw.net

Date: 4-1-05

"EXPRESS MAIL" mailing label number EO 004 396 091 US. Date of Deposit 4/4/05. I hereby certify that this paper is being deposited with the U.S. Postal Service Express Mail Post Office to Addressee Service under 37 C.F.R. §1.10 on the date shown above and is addressed to the U.S. Commissioner of Patents and Trademarks, Alexandria, VA, 22313.

By: _____